IN THE CLAIMS

Please amend the claims as follows:

1-13. (Canceled)

14. (Currently Amended) A computer-implemented method comprising:

~~providing~~ generating a control flow graph of a program, the graph having an inner region with multiple entry nodes, and an outer region;

selecting a single one of the existing entry nodes as a representative entry node for the inner region;

replacing the inner region with the representative entry node;

for each prolog node of the inner region having an edge to one of the existing entry nodes other than the representative entry node, adding an edge from the prolog node to the representative entry node; and

for each epilog node of the inner region, adding an edge from the representative entry node to the epilog node;

assigning edge values to all edges in the control flow graph such that the sum of the edge values along each unique path is unique within the control flow graph.

15. (Previously Presented) The computer-implemented method of claim 14 further comprising:

creating a region source node for the outer region;

for each entry node of the outer region, adding an edge from the region source node to the entry node;

creating a region sink node for the outer region; and

for each exit node of the outer region, adding an edge from the exit node to the region sink node.

16. (Original) The computer-implemented method of claim 15 wherein the control flow graph includes a plurality of inner regions, and the actions of the method are applied for each of the plurality of inner regions, such that a different augmented control flow graph is created for each of the plurality of inner regions.

17. (Original) The computer-implemented method of claim 15 wherein the control flow graph includes a hierarchy of inner regions, and the actions of the method are applied recursively to the hierarchy of inner regions, such that a different augmented control flow graph is created for each inner region in the hierarchy of inner regions.

18. (Canceled)

19. (Previously Presented) The computer-implemented method of claim 23 further comprising:

removing any edges from prolog nodes of the inner region to entry nodes of the inner region other than the representative entry node; and

removing any edges from exit nodes of the inner region other than the representative exit node to epilog nodes of the inner region.

20. (Previously Presented) The computer-implemented method of claim 23 wherein the software function has a function entry and a function exit, and the inner region has at least one entry node and at least one exit node, the method further comprising:

adding an edge from the function entry to each of the at least one entry node of the inner region; and

adding an edge from each of the at least one exit node of the inner region to the function exit.

21. (Previously Presented) The computer-implemented method of claim 23 wherein the control flow graph includes a plurality of inner regions, and the actions of the method are applied for each of the plurality of inner regions.

22. (Previously Presented) The computer-implemented method of claim 23 wherein the control flow graph includes a hierarchy of inner regions, and the actions of the method are applied recursively to the hierarchy of inner regions.

23. (Currently Amended) A computer-implemented method comprising:

in a control flow graph, selecting a representative path within an inner region, having entry and exit nodes that define prolog and epilog nodes, of a software function having a function entry and a function exit, the representative path being identified by a single entry node selected from among multiple existing entry nodes of the inner region, and a single representative exit node selected from among multiple existing exit nodes of the inner region;

adding an edge from the function entry to each entry node of the inner region;

adding an edge from each exit node of the inner region to the function exit;

for each prolog node of the inner region having an edge to one of the existing entry nodes that is not the representative entry node, adding an edge from the prolog node to the representative entry node; and

for each epilog node of the inner region having an edge to one of the existing exit nodes that is not the representative exit node, adding an edge from the representative exit node to the epilog node; and

assigning edge values to all edges in the control flow graph such that the sum of the edge values along each unique path is unique within the control flow graph.

24-39. (Canceled)

40. (Previously Presented) A method comprising:

dividing the CFG of a program into a plurality of hierarchical regions, at least some of the regions having multiple entry nodes and/or multiple exit nodes;

selecting one of the regions;

replacing an inner region with respect to the selected region by a representative node;

adding source and sink nodes to the CFG of the one region;

if the one region has multiple entry nodes, adding an edge from the source node to all the entry nodes of the one region;

if the one region has multiple exit nodes, adding an edge from all the exit nodes of the one region to the sink node;

assigning identifiers to the edges in the one region, including the added edges so as to produce a unique combination of the identifiers for each path in the region, including paths from the source node and to the sink node.

41. (Previously Presented) The method of claim 40 where the identifiers are numbers.

42. (Previously Presented) The method of claim 41 where the combination is a sum.

43. (Previously Presented) The method of claim 40 further comprising instrumenting at least some of the edges of the CFG.

44. (Previously Presented) The method of claim 40 further comprising:

executing code within the region, entering the region at one of the entry nodes, and exiting the region at one of the exit nodes;

combining the identifiers of those edges executed within the region to produce the combination.

45. (Previously Presented) The method of claim 44 further comprising initializing the combination after code execution reaches the one entry node from outside the region.

46. (Previously Presented) The method of claim 45 where

the region has multiple entry nodes, only one of which is executed during a single execution of the region;

the combination is initialized to different ones of the identifiers, depending upon which of the multiple entry nodes is executed during the single execution.

47. (Previously Presented) The method of claim 44 further comprising storing the combination upon exiting the region from the one exit node.

48. (Previously Presented) The method of claim 47 where the region has multiple exit nodes, only one of which is executed during a single execution of the region.

49. (Currently Amended) A machine-readable medium bearing instructions executable by a computer for carrying out the method of:

dividing the CFG of a program into a plurality of hierarchical regions, at least ~~some~~ one of the regions having multiple entry nodes and/or multiple exit nodes;

selecting one of the regions;

replacing an inner region with respect to the selected region by a representative node;

adding source and sink nodes to the CFG of the one region;

if the one region has multiple entry nodes, adding an edge from the source node to all the entry nodes of the one region;

if the one region has multiple exit nodes, adding an edge from all the exit nodes of the one region to the sink node;

assigning identifiers to the edges in the one region, including the added edges so as to produce a unique combination of the identifiers for each path in the region, including paths from the source node and to the sink node.

50. (Currently Amended) A method comprising:

~~providing~~ generating a CFG of a program for profiling, the program having a number of existing edges, a function entry node, and a function exit node;

dividing the CFG into multiple hierarchical regions comprising outer and inner regions each having at least one region entry node and at least one region exit node;

selecting representative paths within the regions;

inserting additional edges into the CFG to produce an augmented CFG, at least ~~some~~ one of the edges extending between an outer one of the regions and an inner one of the regions;

assigning edge identifiers to the existing edges and to at least some of the additional edges such that any path, including paths through the some additional edges, in the augmented CFG has a unique combination of edge identifiers.

51. (Previously Presented) The method of claim 50 where the identifiers are numbers.

52. (Previously Presented) The method of claim 51 where the combination is a sum.

53. (Previously Presented) The method of claim 50 further comprising, for at least some of the regions having multiple paths therethrough, selecting one of the multiple paths as a representative path.

54. (Previously Presented) The method of claim 50 further comprising removing backedges from the regions.

55. (Previously Presented) The method of claim 50 where inserting includes inserting additional edges from the function entry node to the region entry node(s) of the hierarchical regions.

56. (Previously Presented) The method of claim 55 where inserting includes inserting additional edges to all of the region entry nodes of those regions having multiple region entry nodes.

57. (Previously Presented) The method of claim 50 where inserting includes inserting additional edges from the function exit node(s) of the hierarchical regions to the function exit node.

58. (Previously Presented) The method of claim 57 where inserting includes inserting additional edges from all of the region exit nodes of those regions having multiple region exit nodes.

59. (Previously Presented) The method of claim 50 further comprising selecting a single representative entry node for each of the regions having multiple entry nodes.

60. (Previously Presented) The method of claim 59 where inserting includes, for each region, inserting additional edges from prolog nodes of the each region to the representative entry node of that region.

61. (Previously Presented) The method of claim 50 further comprising selecting a single representative exit node for each of the regions having multiple exit nodes.

62. (Previously Presented) The method of claim 61 where inserting includes, for each region, inserting additional edges from the representative exit node to the epilog nodes of the each region.

63. (Previously Presented) The method of claim 50 further comprising instrumenting the existing edges of the augmented CFG.

64. (Previously Presented) The method of claim 50 further comprising:

executing code so as to follow one of multiple plain paths through the program;

combining the identifiers of those edges executed within the program to produce the combination.

65. (Previously Presented) The method of claim 64 further comprising:

accessing one of an array of counters specified by the combination representing the one plain path;

modifying the one counter.

66. (Previously Presented) The method of claim 65 where the number of counters is less than the possible number of plain paths through the program.

67. (Currently Amended) A machine-readable medium bearing instructions executable by a computer for carrying out the method of:

~~providing~~ generating a CFG of a program for profiling, the program having a number of existing edges, a function entry node, and a function exit node;

dividing the CFG into multiple hierarchical regions comprising outer and inner regions each having at least one region entry node and at least one region exit node;

selecting representative paths within the regions;

inserting additional edges into the CFG to produce an augmented CFG, at least ~~some~~ one of the edges extending between an outer one of the regions and an inner one of the regions;

assigning edge identifiers to the existing edges and to at least some of the additional edges such that any path, including paths through the some additional edges, in the augmented CFG has a unique combination of edge identifiers.

68. (Currently Amended) A machine-readable medium bearing instructions executable by a computer for carrying out the method of:

~~providing~~ generating a control flow graph of a program, the graph having an inner region with multiple entry nodes and defining prolog and epilog nodes, and an outer region;

selecting a single one of the existing entry nodes as a representative entry node for the inner region;

replacing the inner region with the representative entry node;

for each prolog node of the inner region having an edge to one of the existing entry nodes other than the representative entry node, adding an edge from the prolog node to the representative entry node; and

for each epilog node of the inner region, adding an edge from the representative entry node to the epilog node;

assigning edge values to all edges in the control flow graph such that the sum of the edge values along each unique path is unique within the control flow graph.

69. (Currently Amended) A machine-readable medium bearing instructions executable by a computer for carrying out the method of:

in a control flow graph, selecting a representative path within an inner region, having entry and exit nodes that define prolog and epilog nodes, of a software function having a function entry and a function exit, the representative path being identified by a single entry node selected from among multiple existing entry nodes of the inner region, and a single representative exit node selected from among multiple existing exit nodes of the inner region;

adding an edge from the function entry to each entry node of the inner region;

adding an edge from each exit node of the inner region to the function exit;

for each prolog node of the inner region having an edge to one of the existing entry nodes that is not the representative entry node, adding an edge from the prolog node to the representative entry node; and

for each epilog node of the inner region having an edge to one of the existing exit nodes that is not the representative exit node, adding an edge from the representative exit node to the epilog node; and

assigning edge values to all edges in the control flow graph such that the sum of the edge values along each unique path is unique within the control flow graph.